Evaluation of Integrals via Numerical Computations

David H. Bailey
Lawrence Berkeley National Laboratory
http://crd.lbl.gov/~dhbailey

PSLQ

Let (x_n) be a vector of real numbers. An integer relation algorithm finds integers (a_n) such that

$$a_1 x_1 + a_2 x_2 + \cdots + a_n x_n = 0$$

- The "PSLQ" algorithm of mathematician-sculptor Helaman Ferguson is the best-known integer relation algorithm.
- PSLQ was named one of ten "algorithms of the century" by *Computing in Science and Engineering*.
- High precision arithmetic software is required: at least d x n digits, where d is the size (in digits) of the largest of the integers a_k.

The BBP Algorithm for Pi

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k} \left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right)$$

- Discovered by a computer running PSLQ in 1996.
- Permits one to directly calculate binary or hexadecimal digits beginning at the n-th position, without needing to compute any of the first n-1 digits.
- Has led to a partial result on the long-standing question of whether pi is 2-normal. [Note: alpha is 2-normal if all m-long binary strings appear in the binary expansion of alpha with limiting frequency 2^{-m}.]
- No similar formula exists for non-binary bases (proven in 2004 by Borwein, Borwein and Galway).

History of Numerical Quadrature

- 1670: Newton devises Newton-Coates integration.
- 1740: Thomas Simpson develops Simpson's rule.
- 1820: Gauss develops Gaussian quadrature.
- 1900s: Adaptive quadrature, Romberg integration, Clenshaw-Curtis integration, others.
- 1990s: Maple and Mathematica feature built-in numerical quadrature facilities.
- 2000s: Very high-precision quadrature (1000+ digits).

Example 1 (2002)

Let

$$C(a) = \int_0^1 \frac{\arctan\sqrt{x^2+a^2}}{(x^2+1)\sqrt{x^2+a^2}}dx$$

Then

$$C(0) = (\pi \log 2)/8 + G/2$$

$$C(1) = \pi/4 - \pi\sqrt{2}/2 + 3\sqrt{2}/2 \cdot \arctan\sqrt{2}$$

$$C(\sqrt{2}) = 5\pi^2/96$$

Several general results have now been proven, including

$$\int_0^\infty \frac{\arctan\sqrt{x^2+a^2}\,dx}{(x^2+1)\sqrt{x^2+a^2}} = \frac{\pi}{2\sqrt{a^2-1}}\left(2\arctan\sqrt{a^2-1} - \arctan\sqrt{a^4-1}\right)$$

Example 2 (2004)

$$\frac{2}{\sqrt{3}}\int_0^1 \frac{\log^6 x \arctan[x\sqrt{3}/(x-2)]}{x+1}\,dx \stackrel{?}{=}$$

$$\frac{1}{81648}\big(-229635L_3(8)+29852550L_3(7)\log 3$$

$$-1632960L_3(6)\pi^2+27760320L_3(5)\zeta(3)$$

$$-275184L_3(4)\pi^4+36288000L_3(3)\zeta(5)$$

$$-30008L_3(2)\pi^6-57030120L_3(1)\zeta(7)\big)$$

where

$$L_3(s) = \sum_{n=1}^{\infty}\left[1/(3n-2)^s - 1/(3n-1)^s\right]$$

is the Dirichlet series.

Example 3 (Sept 2004)

$$\frac{24}{7\sqrt{7}} \int_{\pi/3}^{\pi/2} \log \left| \frac{\tan t + \sqrt{7}}{\tan t - \sqrt{7}} \right| dt \stackrel{?}{=} L_{-7}(2)$$

where $L_{-7}(s)$ is the Dirichlet series

$$L_{-7}(s) = \sum_{n=0}^{\infty} \left[\frac{1}{(7n+1)^s} + \frac{1}{(7n+2)^s} - \frac{1}{(7n+3)^s} + \frac{1}{(7n+4)^s} - \frac{1}{(7n+5)^s} - \frac{1}{(7n+6)^s} \right]$$

The above "identity" has been verified numerically to over 2000 digits, but no proof is known.

Note that the integrand function has a nasty singularity at t = arctan[sqrt(7)].

Example 4 (Sept 2004)

Define

$$J_n = \int_{n\pi/60}^{(n+1)\pi/60} \log \left| \frac{\tan t + \sqrt{7}}{\tan t - \sqrt{7}} \right| dt$$

Then

$$\begin{aligned} 0 \stackrel{?}{=} & -2J_2 - 2J_3 - 2J_4 + 2J_{10} + 2J_{11} + 3J_{12} \\ & +3J_{13} + J_{14} - J_{15} - J_{16} - J_{17} - J_{18} \\ & -J_{19} + J_{20} + J_{21} - J_{22} - J_{23} + 2J_{25} \end{aligned}$$

This has been verified to over 1000 digits. Note as before that the interval in J_{23} includes a nasty singularity.

Example 5 (2003)

$$\frac{2}{3}\int_0^1\int_0^1\sqrt{x^2+y^2}\,dx\,dy+\frac{1}{3}\int_0^1\int_0^1\sqrt{1+(u-v)^2}\,du\,dv$$
$$=\frac{1}{9}\sqrt{2}+\frac{5}{9}\log(1+\sqrt{2})+\frac{2}{9}$$

Quadrature and Experimental Mathematics

These and similar results suggest the need for a general purpose, high-precision quadrature tool:

- Works for most “reasonable” C-infinity functions on finite intervals.
- Must be able to handle vertical derivatives or blow-up singularities at endpoints.
- Must be able to handle integrals on an infinite interval.
- Achieves accuracy to 100s or 1000s of digits accuracy, yet runs in reasonable time on modern computer systems.
- It would be nice do this for 2-D or 3-D integrals also.

Why Not Just Use Mathematica or Maple?

Maple and Mathematica are able to symbolically evaluate each of these three integrals:

$$
\begin{aligned}
I_1 &= \int_0^1 \frac{t^2 \ln(t)\, dt}{(t^2-1)(t^4+1)} \\
I_2 &= \int_0^{\pi/4} \frac{t^2\, dt}{\sin^2(t)} \\
I_3 &= \int_0^{\pi} \frac{x \sin x\, dx}{1+\cos^2 x}
\end{aligned}
$$

However the expressions they produce are quite complicated. For example, the result produced by Mathematica 5.1 for I_3 is over 30 lines long.

Experimental Solutions to the Three Problems

Using a numerical quadrature program and a PSLQ-based online constant recognition facility, we find that

$$\begin{aligned} I_1 &= \pi^2(2-\sqrt{2})/32 \\ I_2 &= -\pi^2/16 + \pi \ln(2)/4 + G \\ I_3 &= \pi^2/4 \end{aligned}$$

Here G is Catalan's constant:

$$G = \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)^2} = 0.915965594177219\ldots$$

High-Precision Gaussian Quadrature

Gaussian quadrature approximates an integral as

$$\int_{-1}^{1} f(x)\,dx \approx \sum_{j=0}^{n} w_j f(x_j)$$

Here x_j are the roots of the Legendre polynomial $P_n(x)$ and the weights w_j are given by

$$w_j = \frac{-2}{(n+1)P_n'(x_j)P_{n+1}(x_j)}$$

Values of $P_n(x)$ can be computed using the recurrence

$$\begin{aligned} P_0(x) &= 0 \qquad P_1(x) = 1 \\ (k+1)P_{k+1}(x) &= (2k+1)xP_k(x) - kP_{k-1}(x) \end{aligned}$$

The abscissas and weights can be pre-computed.

The Euler-Maclaurin Formula

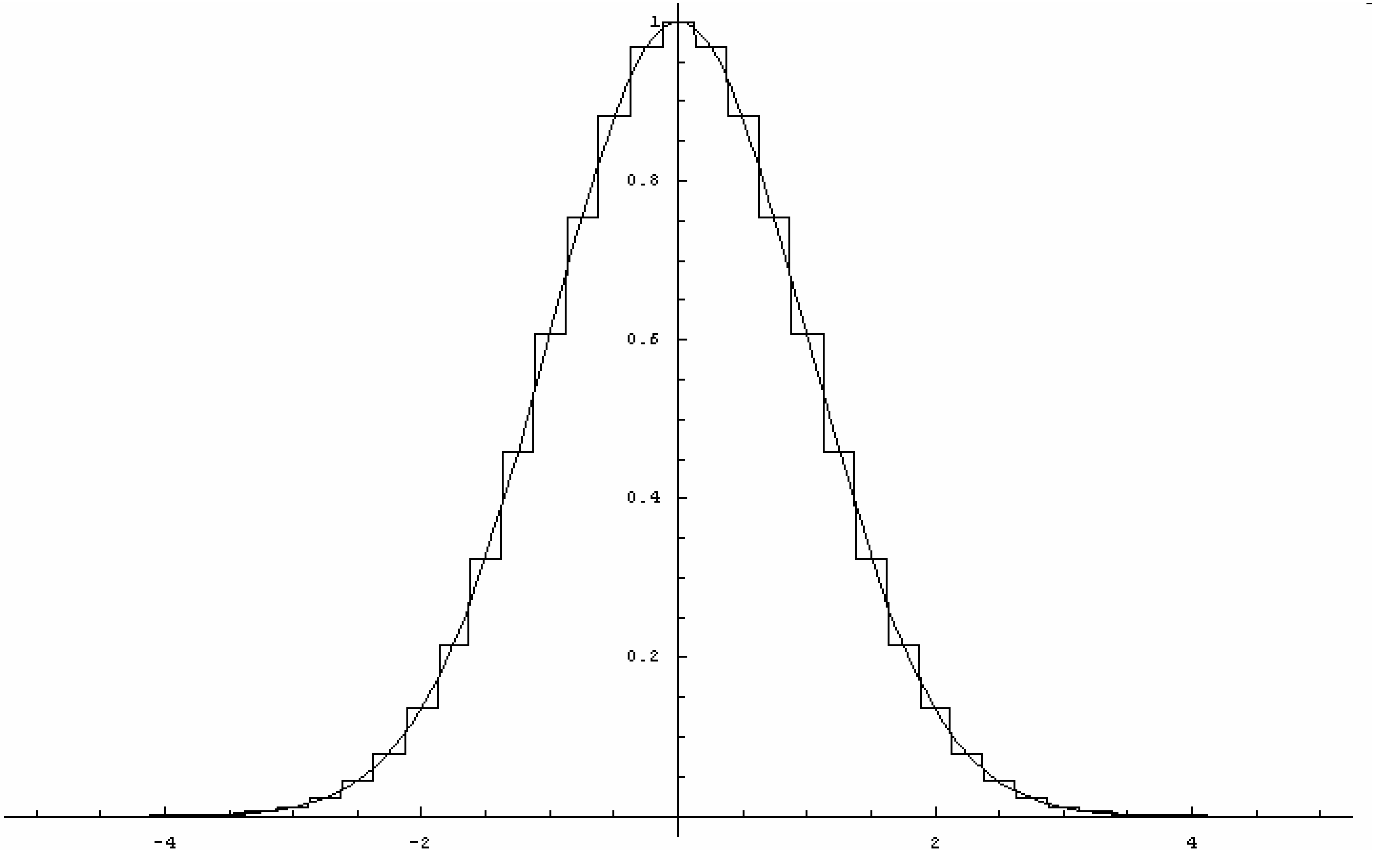
Several "new" numerical integration schemes are based on a remarkable property of certain C-infinity bell-shaped functions, due to the Euler-Maclaurin formula:

$$\begin{aligned}\int_a^b f(x)\,dx &= h\sum_{j=0}^{n} f(x_j) - \frac{h}{2}(f(a)+f(b)) \\ &- \sum_{i=1}^{m} \frac{h^{2i}B_{2i}}{(2i)!}\left(f^{(2i-1)}(b) - f^{(2i-1)}(a)\right) - E(h) \\ |E(h)| &\le 2(b-a)[h/(2\pi)]^{2m+2} \max_{a\le x\le b} |D^{2m+2}f(x)|\end{aligned}$$

[Here $h = (b - a)/n$ and $x_j = a + j\,h$. $D^m f(x)$ means m-th derivative of f.]

Note when f(t) and all of its derivatives are zero at a and b, the error E(h) of a simple block-function approximation to the integral goes to zero more rapidly than any power of h.

Block-Function Approximation to the Integral of a Bell-Shaped Function



Quadrature and the Euler-Maclaurin Formula

Given f(x) defined on (-1,1), employ a function g(t) such that g(t) goes from -1 to 1 over the real line, with g'(t) going to zero for large |t|. Then x = g(t) yields

$$\begin{aligned}\int_{-1}^{1} f(x)\,dx &= \int_{-\infty}^{\infty} f(g(t))g'(t)\,dt \\ &\approx h\sum_{-N}^{N} g'(hj)f(g(hj)) = h\sum_{-N}^{N} w_j f(x_j)\end{aligned}$$

[Here x_j = g(hj) and w_j = g'(hj).]

If g'(t) goes to zero rapidly enough for large t, then even if f(x) has a vertical derivative or blow-up singularity at an endpoint, the product f(g(t))g'(t) often is a nice bell-shaped function for which the E-M formula applies.

Three Suitable 'g' Functions

$$g(t) = \operatorname{erf}(t) \qquad g'(t) = \frac{2}{\sqrt{\pi}} e^{-t^2}$$

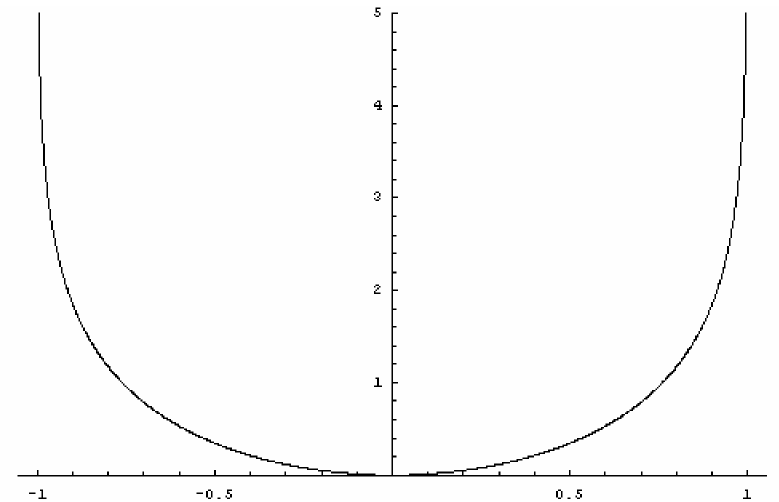
$$g(t) = \tanh t \qquad g'(t) = \frac{1}{\cosh^2 t}$$

$$g(t) = \tanh(\pi/2 \cdot \sinh t) \qquad g'(t) = \frac{\pi/2 \cdot \sinh t}{\cosh^2(\pi/2 \cdot \sinh t)}$$

Original and Transformed Integrand Function

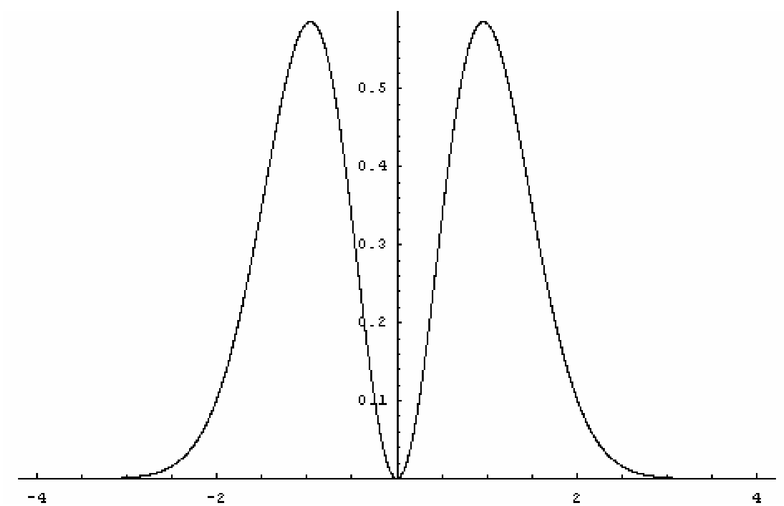
Original function (on [-1,1]):

$$f(t) \quad = \quad -\log\cos\left(\frac{\pi t}{2}\right)$$



Transformed function (on R):

$$f(g(t))g'(t) \quad = \quad -\frac{2}{\sqrt{\pi}}\log\cos\left(\frac{\pi \operatorname{erf} t}{2}\right)\exp(-t^2)$$



Test Integrals

Continuous, well-behaved integrals:

$$1: \int_0^1 t \log(1+t)\, dt = 1/4 \qquad 2: \int_0^1 t^2 \arctan t\, dt = (\pi - 2 + 2\log 2)/12$$

$$3: \int_0^{\pi/2} e^t \cos t\, dt = (e^{\pi/2} - 1)/2 \qquad 4: \int_0^1 \frac{\arctan(\sqrt{2+t^2})}{(1+t^2)\sqrt{2+t^2}}\, dt = 5\pi^2/96$$

Continuous functions with a vertical derivative at endpoint:

$$5: \int_0^1 \sqrt{t} \log t\, dt = -4/9 \qquad 6: \int_0^1 \sqrt{1-t^2}\, dt = \pi/4$$

Functions with a blow-up singularity at an endpoint:

$$7: \int_0^1 \frac{t}{\sqrt{1-t^2}} dt = 1 \qquad 8: \int_0^1 \log t^2\, dt = 2$$

$$9: \int_0^{\pi/2} \log(\cos t)\, dt = -\pi \log(2)/2 \qquad 10: \int_0^{\pi/2} \sqrt{\tan t}\, dt = \pi\sqrt{2}/2$$

Test Integrals, Cont.

Functions on an infinite interval:

$$11: \int_0^\infty \frac{1}{1+t^2}\,dt = \pi/2 \qquad 12: \int_0^\infty \frac{e^{-t}}{\sqrt{t}}\,dt = \sqrt{\pi}$$

$$13: \int_0^\infty e^{-t^2/2}\,dt = \sqrt{\pi/2}$$

Oscillatory function on an infinite interval:

$$14: \int_0^\infty e^{-t}\cos t\,dt = 1/2$$

LBNL's Arbitrary Precision Computation (ARPREC) Software

- Written in C++ for performance and portability.
- C++ and Fortran-90 translation modules that permit conventional C++ and Fortran-90 programs to utilize the package with only very minor changes.
- Arbitrary precision integer, floating and complex datatypes.
- Support for datatypes with differing precision levels.
- Common transcendental functions (exp, sin, log, erf, etc).
- Quadrature routines.
- PSLQ routines.
- Special routines for extra-high precision (>1000 digits).

Available at: **http://www.expmath.info**

1-D Test Results (1000 Digits)

	QUADGS			QUADERF			QUADTS		
Prob.	Level	Time	Error	Level	Time	Error	Level	Time	Error
Init	12	73,046.28		13	3,975.56		12	387.07	
1	7	6.86	10^{-1012}	10	123.19	10^{-1006}	9	39.53	10^{-1011}
2	7	9.13	10^{-1011}	11	147.03	10^{-1006}	9	33.85	10^{-1011}
3	7	10.01	10^{-1009}	10	135.99	10^{-1005}	9	43.51	10^{-1009}
4	7	9.31	10^{-1010}	11	620.60	10^{-1007}	9	72.97	10^{-1010}
5	12	14.70	10^{-13}	10	128.50	10^{-1006}	8	20.17	10^{-1001}
6	12	1.39	10^{-15}	10	6.97	10^{-1006}	9	2.27	10^{-1010}
7	12	1.66	10^{-5}	10	9.74	10^{-1002}	9	2.85	10^{-1003}
8	12	13.89	10^{-8}	10	123.76	10^{-1005}	8	19.45	10^{-1010}
9	12	18.66	10^{-9}	10	139.48	10^{-1005}	9	48.46	10^{-1009}
10	12	7.06	10^{-5}	10	37.20	10^{-1001}	9	15.73	10^{-1002}
11	8	0.41	10^{-1012}	11	10.21	10^{-1006}	10	2.79	10^{-1010}
12	12	7.98	10^{-5}	13	238.98	10^{-1001}	11	80.33	10^{-1002}
13	11	98.50	10^{-1011}	13	158.26	10^{-1005}	12	110.78	10^{-1010}
14	12	6.64	10^{-24}	13	234.29	10^{-876}	12	184.59	10^{-1007}

Quadratic Convergence in Erf Quadrature

Level	Prob. 2	Prob. 4	Prob. 6	Prob. 8	Prob. 10	Prob. 12	Prob. 14
1	10^{-1}	10^{-1}	10^{-1}	10^{0}	10^{-1}	10^{0}	10^{0}
2	10^{-2}	10^{-5}	10^{-3}	10^{-3}	10^{-3}	10^{-1}	10^{-1}
3	10^{-6}	10^{-6}	10^{-8}	10^{-10}	10^{-8}	10^{-3}	10^{-2}
4	10^{-13}	10^{-12}	10^{-17}	10^{-21}	10^{-16}	10^{-6}	10^{-3}
5	10^{-26}	10^{-25}	10^{-34}	10^{-43}	10^{-33}	10^{-11}	10^{-5}
6	10^{-52}	10^{-51}	10^{-68}	10^{-87}	10^{-66}	10^{-20}	10^{-10}
7	10^{-104}	10^{-102}	10^{-134}	10^{-173}	10^{-132}	10^{-37}	10^{-19}
8	10^{-206}	10^{-204}	10^{-266}	10^{-348}	10^{-264}	10^{-70}	10^{-37}
9	10^{-411}	10^{-409}	10^{-529}	10^{-696}	10^{-527}	10^{-132}	10^{-68}
10	10^{-821}	10^{-819}	10^{-1004}	10^{-1004}	10^{-1001}	10^{-249}	10^{-128}
11	10^{-1003}	10^{-1003}				10^{-472}	10^{-242}
12						10^{-896}	10^{-460}
13						10^{-1001}	10^{-876}

Each successive level halves the size of h, thus doubling the number of abscissa-weight pairs (and function evaluations).

Tanh-sinh quadrature also exhibits quadratic convergence.

Performance of Parallel 1-D Tanh-Sinh Program (2000-Digit Runs)

Problem Number	Levels Required	Processors 1	4	16	64	256	1024
Init		22462.17	5668.95	1439.76	360.53	92.79	25.92
1	10	1952.09	499.64	125.95	31.98	8.34	3.19
2	10	5575.73	1433.29	363.06	92.46	24.65	7.85
3	10	2865.02	732.78	186.25	46.51	12.46	4.00
4	10	6220.04	1596.34	403.42	103.33	27.26	8.43
5	9	986.87	254.09	64.00	16.48	4.58	1.42
6	10	105.40	27.21	6.85	1.75	0.48	0.26
7	10	223.78	58.06	14.40	3.76	0.95	0.33
8	9	975.23	249.93	63.94	16.42	4.56	1.42
9	10	3078.12	790.60	201.44	51.32	13.28	4.03
10	10	1377.10	361.05	91.28	23.65	6.09	1.97
11	11	91.37	23.45	6.00	1.55	0.42	0.24
12	12	3305.49	838.17	211.60	53.53	13.82	4.21
13	13	4469.49	1136.02	284.50	71.78	18.55	5.00
14	13	13960.36	3595.45	907.79	231.07	59.12	15.50
Total		67648.26	17265.03	4370.24	1106.12	287.35	83.77
Ratio		1.00	3.92	15.48	61.16	235.42	807.55

2-D Tanh-Sinh Quadrature

The 1-D tanh-sinh quadrature scheme can be generalized to 2-D integrals as follows:

$$\begin{aligned} \int_{-1}^{1}\int_{-1}^{1} f(x,y)\,dx\,dy &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(g(s),g(t))g'(s)g'(t)\,ds\,dt \\ &\approx h\sum_{k=-N}^{N}\sum_{j=-N}^{N} w_j w_k f(x_j,x_k) \end{aligned}$$

where as before

$$\begin{aligned} g(t) &= \tanh(\pi/2\cdot\sinh(t)) \\ g'(t) &= \pi/2\cdot\sinh t/\cosh^2(\pi/2\cdot\sinh t) \\ x_j &= g(hj) \\ w_j &= g'(hj) \end{aligned}$$

2-D Test Problems

$$1 : \int_0^1 \int_0^1 \sqrt{s^2+t^2}\, ds\, dt = \sqrt{2}/3 - \log(2)/6 + \log(2+\sqrt{2})/3$$

$$2 : \int_0^1 \int_0^1 \sqrt{1+(s-t)^2}\, ds\, dt = -\sqrt{2}/3 - \log(\sqrt{2}-1)/2 + \log(\sqrt{2}+1)/2 + 2/3$$

$$3 : \int_{-1}^1 \int_{-1}^1 (1+s^2+t^2)^{-1/2}\, ds\, dt = 4\log(2+\sqrt{3}) - 2\pi/3$$

$$4 : \int_0^\pi \int_0^\pi \log[2-\cos s - \cos t]\, ds\, dt = 4\pi G - \pi^2 \log 2$$

$$5 : \int_0^\infty \int_0^\infty \sqrt{s^2+st+t^2}\, e^{-s-t}\, ds\, dt = 1 + 3/4 \cdot \log 3$$

$$6 : \int_0^1 \int_0^1 (s+t)^{-1}[(1-s)(1-t)]^{-1/2}\, ds\, dt = 4G$$

$$7 : \int_0^1 \int_0^t (1+s^2+t^2)^{-1/2}\, ds\, dt = -\pi/12 - 1/2 \cdot \log 2 + \log(1+\sqrt{3})$$

$$8 : \int_0^\pi \int_0^t (\cos s \sin t) e^{-s-t}\, ds\, dt = 1/4 \cdot (1+e^{-\pi})$$

Performance of 2-D Parallel Tanh-Sinh Program (120-Digit Runs)

Problem Number	Levels Required	Processors 1	16	64	256	1024
1	9	4854.89	384.60	98.76	25.53	10.51
2	6	72.56	5.96	1.70	0.97	2.43
3	7	328.90	26.21	6.91	2.36	3.17
4	9	60475.50	4826.64	1228.59	307.82	82.08
5	9	8973.73	696.87	177.61	45.26	14.93
6	9	6448.27	495.78	127.50	32.78	11.60
7	6	91.13	7.34	2.04	1.04	2.45
8	6	449.40	36.10	9.66	3.08	3.13
Total		81694.38	6479.50	1652.77	418.84	130.30
Ratio		1.00	12.61	49.43	195.05	626.97

All results were correct to over 100 digits, except for Problem 4 (10^{-86}) and Problem 6 (10^{-80}) – one more level would be needed to achieve over 100-digit accuracy on these two.

1-D vs 2-D and 3-D

- 2-D is much more expensive than 1-D, because many more function evaluations are required at a given spacing.
- For 1-D, each additional level doubles the number of function evaluations required; for 2-D, it is quadrupled.
- 1-D exhibits quadratic convergence on many problems; for 2-D, only for functions that are well-behaved at boundary; otherwise about 1.4X per level.
- 3-D quadrature is possible, but many times more expensive than 2-D.

Caution: Example 1

$$\int_0^\infty \frac{\sin x}{x} dx = \frac{\pi}{2}$$

$$\int_0^\infty \frac{\sin x}{x} \cdot \frac{\sin(x/3)}{x/3} dx = \frac{\pi}{2}$$

$$\int_0^\infty \frac{\sin x}{x} \cdot \frac{\sin(x/3)}{x/3} \cdot \frac{\sin(x/5)}{x/5} dx = \frac{\pi}{2}$$

$$\cdots$$

$$\int_0^\infty \frac{\sin x}{x} \cdot \frac{\sin(x/3)}{x/3} \cdot \frac{\sin(x/5)}{x/5} \cdots \frac{\sin(x/13)}{x/13} dx = \frac{\pi}{2}$$

but

$$\int_0^\infty \frac{\sin x}{x} \cdot \frac{\sin(x/3)}{x/3} \cdot \frac{\sin(x/5)}{x/5} \cdots \frac{\sin(x/15)}{x/15} dx$$

$$= \frac{467807924713440738696537864469}{935615849440640907310521750000}\pi$$

Caution: Example 2

These constants agree to 42 decimal digit accuracy, but are NOT equal:

$$\int_0^\infty \cos(2x) \prod_{n=0}^{\infty} \cos(x/n)\, dx \quad =$$

$$0.392699081698724154807830422909937860524645434187231\ldots$$

$$\frac{\pi}{8} \quad =$$

$$0.392699081698724154807830422909937860524646174921888\ldots$$

Computing this integral is nontrivial, due to the difficulty in evaluating the integrand function to high precision. See 2004 manuscript by DHB, Jon Borwein, Vishaal Kapoor and Eric Weisstein for details.

Open Questions

- Prove various numerical discoveries such as

$$\frac{24}{7\sqrt{7}} \int_{\pi/3}^{\pi/2} \log \left| \frac{\tan(t) + \sqrt{7}}{\tan(t) - \sqrt{7}} \right| dt \stackrel{?}{=} L_{-7}(2)$$

- Understand why and when the erf and tanh-sinh schemes achieve quadratic convergence on 1-D problems. Why 2X?
- Understand why 2-D tanh-sinh exhibits quadratic convergence on "nice" problems, but lower rates on others.
- Is there a fundamentally better way to do 1-D? 2-D? 3-D?
- Better understand what classes of functions have closed-form definite integrals.
- Find a more systematic way to guess the form of terms in possible evaluations, as input to PSLQ.